Attorney Docket
13644.0046

Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

IN THE CLAIMS:

1. (Original) A method for executing a computer program having source code on a target computer platform having a memory, the method comprising the steps of:

defining a plurality of program objects for the computer program;

placing a first plurality of directives in the source code to divide the computer program into the program objects, whereby an annotated computer program is produced;

processing the annotated computer program to generate a description for each of the program objects;

allocating the program objects to fixed locations in the memory of the target computer platform;

porting the annotated computer program to the target computer platform;

generating an executable image of the annotated computer program, wherein the executable image is configured for execution on the target computer platform; and

executing the executable image on the target computer platform.

2. (Original) The method according to claim 1, wherein in the step of defining a plurality of program objects, the program objects comprise executable code, constant data, and volatile data.

3. (Original) The method according to claim 1 further comprising the step of estimating a typical usage for each of the program objects.

4. (Original) The method according to claim 1 further comprising the step of designating each of the program objects as one of (a) a static program object and (b) an overlay program object.

5. (Original) The method according to claim 1, wherein the step of porting the annotated computer program further comprises binding each of the directives to an object management system of the target computer platform.

6. (Original) The method according to claim 1 further comprising the step of placing a second plurality of directives in the source code to indicate linkages between program objects.

7. (Original) The method according to claim 1 further comprising the step of:

identifying a plurality of natural application boundaries in the source code; and

wherein the step of placing a first plurality of directives further comprises placing the first plurality of directives in the source code at the natural application boundaries.

8. **(Amended)** The method according to claim 1, wherein in the step of defining a plurality of program objects, each of the program objects has a unique ~~name~~ **name.**

9. (Original) A computer program product embodied on a first computer for facilitating the execution of a computer program having source code on a second computer having a memory, the computer program product comprising:

a compiler code segment comprising computer readable program code configured to cause the first computer to perform the steps of:

defining a plurality of program objects for the computer program; and

placing a first plurality of directives in the source code to divide the computer program into the program objects, whereby an annotated computer program is produced;

an extraction code segment comprising computer readable program code configured to cause the first computer to process the annotated computer program to generate a description for each of the program objects;

an object allocation code segment comprising computer readable program code configured to cause the first computer to allocate the program objects to fixed locations in the memory of the second computer;

a porting code segment comprising computer readable program code configured to cause the first computer to port the annotated computer program to the second computer; and

a merging code segment comprising computer readable program code configured to cause the first computer to generate an executable image of the annotated computer program, wherein the executable image is configured for execution on the second computer.

10. (Original) The computer program product of claim 9 further comprising a test code segment comprising computer readable program code configured to cause the first computer to estimate a typical usage for each of the program objects.

11. (Previously presented) A system for optimizing source code for execution on a target computer platform having a memory, comprising:

a first module defining a plurality of program objects of the source code;

a second module placing a first plurality of directives in the source code to divide the source code into the program objects, whereby an annotated computer program is produced;

a third module processing the annotated computer program to generate a description for each of the program objects;

a fourth module allocating the program objects to fixed locations in the memory of the target computer platform;

a fifth module porting the annotated computer program to the target computer platform;

a compiler compiling and linking the annotated computer program to generate an executable image, wherein the executable image is configured for execution on the target computer platform; and

a processor executing the executable image on the target computer platform.

12. (Previously presented) The system of claim 11, wherein the first module defining the plurality of program objects further defines executable code objects, constant data objects, and volatile data objects.

13. (Previously presented) The system of claim 11 further comprising a sixth module stimulating each program object with a test vector to determine a usage for each program object.

14. (Previously presented) The system of claim 11 further comprising a seventh module designating each of the program objects as one of (a) a program object that is resident for the duration of the application and (b) a program object that is paged as needed.

15. (Previously presented) The system of claim 11, wherein the fifth module binds each of the directives to an object management system of the target computer platform.

16. (Previously presented) The system of claim 11 wherein the second module places a second plurality of directives in the source code to indicate linkages between program objects.

17. (Previously presented) The system of claim 11 further comprising:

an eighth module identifying a plurality of application boundaries in the source code, wherein each application boundary identifies a transition point between two program objects in the source code; and

wherein the second module places the first plurality of directives in the source code at the application boundaries.

18. **(Amended)** The system of claim 11, wherein **the** first module ensures that each of the program objects has a unique name.

19. (Previously presented) The system of claim 11 wherein the second module selects the first plurality of directives form the group comprising a branch, a call, a close, a def_handle, an endobj, an entry, a get, a get_handle, an obj, an open, a put, and a return.